

# Time-Extended Payoffs for Collectives of Autonomous Agents

Kagan Tumer  
NASA Ames Research Center  
Mailstop 269-3  
Moffett Field, CA 94035  
kagan@ptolemy.arc.nasa.gov

Adrian K. Agogino  
The University of Texas  
ENS 518  
Austin, TX 78712  
agogino@ece.utexas.edu

May 2, 2002

## Abstract

A collective is a set of self-interested agents which try to maximize their own utilities, along with a well-defined, time-extended world utility function which rates the performance of the entire system. In this paper, we use theory of collectives to design time-extended payoff utilities for agents that are both “aligned” with the world utility, and are “learnable”, i.e., the agents can readily see how their behavior affects their utility. We show that in systems where each agent aims to optimize such payoff functions, coordination arises as a byproduct of the agents selfishly pursuing their own goals. A game theoretic analysis shows that such payoff functions have the net effect of aligning the Nash equilibrium, Pareto optimal solution and world utility optimum, thus eliminating undesirable behavior such as agents working at cross-purposes.

We then apply collective-based payoff functions to the token collection in a gridworld problem where agents need to optimize the aggregate value of tokens collected across an episode of finite duration (i.e., an abstracted version of rovers on Mars collecting scientifically “interesting” rock samples, subject to power limitations). We show that, regardless of the initial token distribution, reinforcement learning agents using collective-based payoff functions significantly outperform both “natural” extensions of single agent algorithms and ~~global reinforcement learning solutions based on “team games”.~~

## 1 Introduction

There are many problems which can only be properly addressed by having a set of autonomous agents act independently and have their *joint* sequence of actions optimize a pre-set utility function. Such systems, called **collectives**, can be formally defined by the following characteristics: there is a collection of self-interested agents which try to maximize their own **payoff utility** functions; and there is a well-defined, time-extended **world utility** function, which rates the performance of the entire system [32, 33]. Furthermore, in general, collectives lack centralized communication or control.

Examples of problems which require a collective-based approach include control of a constellation of satellites, construction of distributed algorithms, routing over a data network, and control of a collection of planetary exploration vehicles (e.g., rovers on Mars, or submersibles under Europa's ice caps). For the collective-based approach to work in such problems, two fundamental issues need to be addressed:

1. the agents need to learn the action *sequence* that will provide good values for their payoff utility functions, i.e., agents need to achieve their own goals; and
2. the agents' learning their own payoff utilities needs to benefit the world utility, i.e., the agents' utilities need to be "aligned" with the world utility so that the agents do not work at cross-purposes, as far as the world utility is concerned.

The first of these issues has been dealt with extensively in the single agent context; there are many reinforcement learning systems [20], (e.g., Q-learners [24]) that have successfully been applied to real world problems [2, 1]. The second problem has received less attention, and generally the solution consists of either each agent receiving the world utility as their payoff utility (e.g., "team" games [3]), or of imposing external mechanisms (e.g., contracts, auctions) that encourage the agents to work together [8, 9, 16, 17].

Addressing these two issues simultaneously, without extensive hand-tailoring is the main challenge in designing collectives<sup>1</sup>. In particular, the design problem can be formulated as follows: Assuming the individual agents are able to maximize their own utility functions (e.g., through reinforcement learning), what set of payoff utilities for the individual agents will, when pursued by those agents, result in high world utility? That is, to induce good collective behavior, we leverage the assumption that our learners are individually fairly good at *what they do*, and focus on determining what it is that they *should* do.

The theory of collectives provides the framework to address such design problems [22, 32]. The crux of that theory concerns the derivation of payoff functions that are both "aligned" with the world utility and are "learnable" (discussed in detail in Section 2). Factoredness (a formalization of the concept of "alignedness") ensures that an action taken by an agent that improves its payoff utility also improves the world utility. Learnability ensures that an agent can discern the effect of its own actions on its own utility and select actions that improve that utility. In Section 2 we summarize how, given a world utility, one can design payoff utilities that maximize learnability while restricted to set of functions that are factored.

As a naturally occurring example, consider the human economy as a collective. The agents are identified with the individuals trying to maximize their payoff utilities (e.g., maximize bank account, advance career). An example of a world utility is the time average of the gross domestic product ("world utility" is not a construction internal to a human economy, but rather something defined from the outside). To achieve high world utility it is imperative that the agents do not work at cross-purposes; otherwise frustrational phenomena like the tragedy of the commons, in which individual avarice works to lower world utility [7], can occur. Modifications to the agents' utility functions via punitive legislation is one way in which such undesirable phenomena can be

<sup>1</sup>Though hand-tailoring the agents' payoff utilities may work in some domains, such systems (i) have to be laboriously modeled; (ii) provide "brittle" global performance; (iii) are not "adaptive" to changing environments; and (iv) generally do not scale well.

avoided<sup>2</sup>. Securities and Exchange Commission (SEC) regulations designed to prevent insider trading can be viewed as a real world example of an attempt to make such a modification to the agents' utilities. For example, a trade that once may have added to your wealth while hurting the economy, may now lead to your prosecution. You are therefore unlikely to make such a trade. As a consequence of this new regulation, your payoff utility and the world utility have become more aligned.

The problem of designing a collective is related to work in many fields beyond multiagent systems, and includes mechanism design, reinforcement learning for adaptive control, computational ecologies, and game theory. However none of these fields directly addresses the inverse problem of how to design the agents' utilities to reach a desirable world utility value in its full generality. This is even true for the field of mechanism design, which while addressing an inverse problem similar to that of the design of collectives, does so only for certain restricted domains, and does not address the "learnability" issue. For example, mechanism design is mostly appropriate when there are pre-specified goals underlying agents' utilities over which "incentives" need to be provided, and when Pareto-optimality (rather than optimization of a world utility) is often the goal [31]. We discuss the connection between the Groves mechanism, the AGV-Arrow mechanism and collective-based design in Section 2.

To date, the theory of collectives has been successfully applied to various domains, including packet routing over a data network [33], the congestion game known as Arthur's El Farol Bar problem [34], and data downloads from a constellation of satellites [30]. In particular, in the routing domain, the collective-based approach achieved performance improvements of a factor of three over the conventional Shortest Path Algorithm (SPA) routing algorithms currently running on the internet [29], and avoided the Braess' routing paradox which plagues the SPA-based systems [22].

In the work described above, agents were concerned with optimizing "rewards" (i.e., utility value of a single time step). In this paper we extend these results to a problem where agents need to optimize a time-extended utility function through selecting sequences of actions. We show that in this significantly more complex domain, agents that use collective-based utilities provided solutions that are significantly superior to agents that either use team games or "natural" utilities. In Section 2, we provide a summary of the theory of collectives and discuss its relationship to mechanism design. In Section 3, we describe the gridworld problem domain and develop a collective based solution to the design of agents' payoff utilities. In Section 4, we present simulation results that show that collective-based solutions significantly outperform both traditional and more "natural" solutions. Finally, in Section 5, by studying the Nash equilibrium of a simple system, we demonstrate how collective-based algorithms achieve performance unattainable by systems using "selfish" payoff functions.

---

<sup>2</sup>In this context we say we "modify" the agents' initial utility ( $f_1$ ) when we add incentives ( $f_2$ ). From an economics perspective where an agent's utility is considered immutable, one would refer to this operation as adding incentives to the agent's utility. Whether one calls the function  $g = f_1 + f_2$  the agents' new utility, or whether the agents' utility is still  $f_1$  but it is now subject to incentives  $f_2$  is nothing but a semantics issue.

## 2 Background: Theory of Collectives

In this section, we summarize the portion of the theory of collectives necessary and sufficient to describe the learning of sequences of actions in a distributed system [27]. Let  $Z$  be an arbitrary vector space whose elements  $\zeta$  give the joint move of all agents in the system (i.e.,  $\zeta$  specifies the full “worldline” consisting of the actions/states of all the agents). The provided **world utility**  $G(\zeta)$ , is a function of the full worldline, and we wish to search for the  $\zeta$  that maximizes  $G(\zeta)$ .

In addition to  $G$ , for each agent  $\eta$ , there is a set of **payoff utility functions**  $\{g_\eta\}$ . The agents will act to improve their individual payoff functions, even though, we, as system designers are only concerned with the value of the world utility  $G$ . To specify all agents other than  $\eta$ , we will use the notation  $\hat{\eta}$ .

### 2.1 Intelligence

We need to have a way to “standardize” utility functions so that the numeric value they assign to a  $\zeta$  only reflects the ranking of  $\zeta$  relative to certain other elements of  $Z$ . We call such a standardization of some arbitrary utility  $U$  for agent  $\eta$  the “**intelligence for  $\eta$  at  $\zeta$  with respect to  $U$** ”. Here we use intelligences that are equivalent to percentiles:

$$\epsilon_U(\zeta : \eta) \equiv \int d\mu_{\zeta_\eta(\zeta')} \Theta[U(\zeta) - U(\zeta')], \quad (1)$$

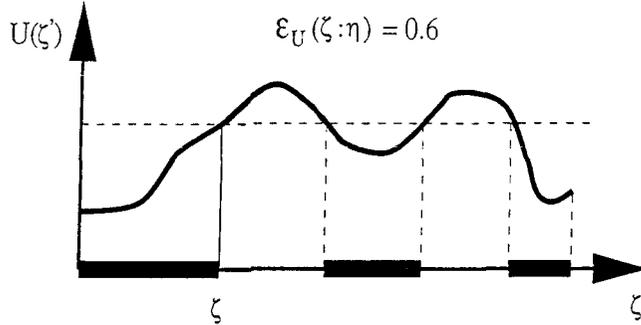
where the Heaviside function  $\Theta$  is defined to equal 1 when its argument is greater than or equal to 0, and to equal 0 otherwise, and where the subscript on the (normalized) measure  $d\mu$  indicates it is restricted to  $\zeta'$  sharing the same non- $\eta$  components as  $\zeta$ .<sup>3</sup> Note that intelligence values are always between 0 and 1. Intuitively, intelligence values indicate what percentage of  $\eta$ 's actions would have resulted in lower utility. Accordingly,  $\epsilon_{g_\eta}(\zeta : \eta) = 1$  means that agent  $\eta$  is fully rational at  $\zeta$ , in that its move maximizes its payoff, given the moves of other agents. Figure 1 shows an example where 60% of  $\eta$ 's actions would have resulted in worse utility, giving  $\eta$  an intelligence of 0.6 at that point ( $\zeta$ ).

Our uncertainty concerning the behavior of the system is reflected in a probability distribution over  $Z$ . Our ability to control the system consists of setting the value of some characteristic of the collection of agents, e.g., setting the payoff functions of the agents. Indicating that value by  $s$ , our analysis revolves around the following central equation for  $P(G | s)$ , which follows from Bayes' theorem:

$$P(G | s) = \int d\vec{\epsilon}_G P(G | \vec{\epsilon}_G, s) \int d\vec{\epsilon}_g P(\vec{\epsilon}_G | \vec{\epsilon}_g, s) P(\vec{\epsilon}_g | s), \quad (2)$$

where  $\vec{\epsilon}_g \equiv (\epsilon_{g_{\eta_1}}(\zeta : \eta_1), \epsilon_{g_{\eta_2}}(\zeta : \eta_2), \dots)$  is the vector of the intelligences of the agents with respect to their associated payoff functions, and  $\vec{\epsilon}_G \equiv (\epsilon_G(\zeta : \eta_1), \epsilon_G(\zeta : \eta_2), \dots)$  is the vector of the intelligences of the agents with respect to  $G$ .

<sup>3</sup>The measure must reflect the type of system at hand, e.g., whether  $Z$  is countable or not, and if not, what coordinate system is being used. Other than that, any convenient choice of measure may be used and the theorems will still hold.



**Figure 1:** Intelligence of agent  $\eta$  at state  $\zeta$  for utility  $U$ :  $\zeta$  is the actual joint move at hand. The x-axis shows agent  $\eta$ 's alternative possible moves (all states  $\zeta'$  having  $\zeta$ 's values for the moves of all agents other than  $\eta$ ). The bold lines show the alternative moves that  $\eta$  could have made that would have given  $\eta$  a worse value of the utility  $U$ . The fraction of those bold lines to the full set of  $\eta$ 's possible moves (which is 0.6 in this example) is the intelligence of agent  $\eta$  at  $\zeta$  for utility  $U$ , denoted by  $\epsilon_U(\zeta : \eta)$ .

Note that, from a game-theoretic perspective, a point  $\zeta$  where all players are rational, ( $\epsilon_{g_\eta}(\zeta : \eta) = 1$  for all agents  $\eta$ ), is a game theory Nash equilibrium [4]. On the other hand, a  $\zeta$  at which all components of  $\vec{\epsilon}_G = 1$  is a local maximum of  $G$  (or more precisely, a critical point of the  $G(\zeta)$  surface).

If we can choose  $s$  so that the third conditional probability in the integrand,  $P(\vec{\epsilon}_g | s)$ , is peaked around vectors  $\vec{\epsilon}_g$  all of whose components are close to 1 (that is agents are able to “learn” their tasks), then we have likely induced large payoff utility intelligences. If we can also have the second term,  $P(\vec{\epsilon}_G | \vec{\epsilon}_g, s)$ , be peaked about  $\vec{\epsilon}_G$  equal to  $\vec{\epsilon}_g$  (that is the payoff and world utilities are aligned), then  $\vec{\epsilon}_G$  will also be large. Finally, if the first term in the integrand,  $P(G | \vec{\epsilon}_G, s)$ , is peaked about high  $G$  when  $\vec{\epsilon}_G$  is large, then our choice of  $s$  will likely result in high  $G$ , as desired.

## 2.2 Factoredness and Learnability

The requirement that payoff functions have high “signal-to-noise” (an issue not considered in conventional work in mechanism design) arises in the third term. It is in the second term that the requirement that the payoff functions be “aligned with  $G$ ” arises. In this work we concentrate on these two terms, and show how to simultaneously set them to have the desired form.<sup>4</sup>

Details of the stochastic environment in which the collection of agents operate, together with details of the learning algorithms of the agents, are reflected in the distribution  $P(\zeta)$  which underlies the distributions appearing in Equation 2. Note though that *independent of these considerations*, our desired form for the second term in Equation 2 is assured if we have chosen payoff utilities such that  $\vec{\epsilon}_g$  equals  $\vec{\epsilon}_G$  exactly for all  $\zeta$ . We call such a system **factored**. In game

<sup>4</sup>Non-game theory-based function maximization techniques like simulated annealing instead address how to have term 1 have the desired form. They do this by trying to ensure that the local maxima that the underlying system ultimately settles near have high  $G$ , by “trading off exploration and exploitation”. One can combine such term-1-based techniques with the techniques presented here. The resultant hybrid algorithm, addressing all three terms, outperforms simulated annealing by over two orders of magnitude [28].

theory language, the Nash equilibria of a factored system are local maxima of  $G$ . In addition to this desirable equilibrium behavior, factored systems also automatically provide appropriate off-equilibrium incentives to the agents (an issue rarely considered in the game theory / mechanism design literature).

As a trivial example, any “team game” in which all the payoff functions equal  $G$  is factored [3, 14]. However team games often have very poor forms for term 3 in Equation 2, forms which get progressively worse as the size of the system grows. This is because for large systems where  $G$ ’s sensitively depends on all components of the system, each agent may experience difficulty discerning the effects of its actions on  $G$ . As a consequence, each  $\eta$  may have difficulty achieving high  $g_\eta$  in a team game. We can quantify this signal/noise effect by comparing the ramifications on  $g_\eta(\zeta)$  arising from changes to  $\zeta_\eta$  with the ramifications arising from changes to  $\zeta_{\neq\eta}$  (i.e., changes to all nodes *other* than  $\eta$ ). We call this quantification the differential **learnability** of payoff utility  $g_\eta$ , in the vicinity of  $\zeta$  [31]:

$$\lambda_{\eta, g_\eta}(\zeta) \equiv \frac{\|\vec{\nabla}_{\zeta_\eta} g_\eta(\zeta)\|}{\|\vec{\nabla}_{\zeta_{\neq\eta}} g_\eta(\zeta)\|}. \quad (3)$$

The denominator in Equation 3 reflects how sensitive  $g_\eta(\zeta)$  is to changes in  $\zeta_{\neq\eta}$ , or changes to agents other than  $\eta$ . In contrast, the numerator reflects how sensitive  $g_\eta(\zeta)$  is to changing  $\zeta_\eta$ . So at a given state  $\zeta$ , the higher the learnability, the more  $g_\eta(\zeta)$  depends on the move of agent  $\eta$ , i.e., the better the associated signal-to-noise ratio for  $\eta$ . Intuitively then, higher learnability means it is easier for  $\eta$  to achieve a large value of its intelligence.

### 2.3 Difference Utilities

It is possible to solve for the set of all payoff utilities that are factored with respect to a particular world utility. Unfortunately, in general it is not possible for a system both to be factored and to have infinite learnability (i.e., no dependence of any  $g_\eta$  on any agent other than  $\eta$ ) for all of its agents [27]. However, consider **difference** utilities, which are of the form:

$$U(\zeta) = G(\zeta) - \Gamma(f(\zeta)), \quad (4)$$

where  $\Gamma(f)$  is independent of  $\zeta_\eta$ . Such difference utilities are factored [31]. In addition, under usually benign approximations, the differential learnability can be maximized over the set of difference utilities by choosing  $f \equiv G$  and setting  $\Gamma$  to the expected value operator [31]. We call the resultant difference utility the **Aristocrat Utility (AU)**, loosely reflecting the fact that it measures the difference between an agent’s actual action and the average action:

$$AU(\zeta) = G(\zeta) - E(G \mid \zeta_\eta, s). \quad (5)$$

Each agent  $\eta$  using its AU as its payoff function, theoretically ensures good form for both terms 2 and 3 in Equation 2. Achieving this however, is not always feasible. The problem is that to evaluate the expectation value defining its AU, each agent needs to evaluate the current probabilities of each of its potential moves. However if the agent then changes its payoff function to be the associated AU, it will in general substantially change its ensuing behavior. (The agent now wants to choose moves that maximize a different function from the one it was maximizing

$$\begin{array}{ccc}
\begin{array}{l} \eta_1 \\ \eta_2 \\ \eta_3 \\ \eta_4 \end{array} & \begin{array}{c} \zeta \\ \left[ \begin{array}{ccc} 1 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array} \right] \end{array} & \begin{array}{c} \Rightarrow \\ \text{Clamp } \eta_2 \\ \text{to "null"} \end{array} & \begin{array}{c} (\zeta_{\eta_2}, \vec{0}) \\ \left[ \begin{array}{ccc} 1 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array} \right] \end{array} \\
& \begin{array}{c} \Rightarrow \\ \text{Clamp } \eta_2 \\ \text{to "average"} \end{array} & & \begin{array}{c} (\zeta_{\eta_2}, \vec{a}) \\ \left[ \begin{array}{ccc} 1 & 0 & 0 \\ \mathbf{.33} & \mathbf{.33} & \mathbf{.33} \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array} \right] \end{array}
\end{array}$$

**Figure 2:** This example shows the impact of clamping on the joint state of a four-agent system where each agent has three possible actions, and each such action is represented by a three-dimensional unary vector. The first matrix represents the joint state of the system  $\zeta$  where agent 1 has selected action 1, agent 2 (shown in bold for emphasis) has selected action 3, agent 3 has selected action 1 and agent 4 has selected action 2. The second matrix displays the effect of clamping agent 2’s action to the “null” vector (i.e., replacing  $\zeta_{\eta_2}$  with  $\vec{0}$ ). The third matrix shows the effect of instead clamping agent 2’s move to the “average” action vector  $\vec{a} = \{.33, .33, .33\}$ , which amounts to replacing that agent’s move with the “illegal” move of fractionally taking each possible move ( $\zeta_{\eta_2} = \vec{a}$ ).

before.) In other words, it will change the probabilities of its moves, which means that its new payoff function is in fact not the AU for its actual (new) probabilities.

There are ways around this self-consistency problem (including setting the probabilities of taking the possible actions to fixed values as discussed in Section 3), but in practice it is often easier to bypass the entire issue, by giving each  $\eta$  a payoff function that does not depend on the probabilities of  $\eta$ ’s own moves. Before defining a family of such payoff utility functions, let us define the **clamping parameter**  $CL_{\eta}^{\vec{v}}$  as the fictitious state where agent  $\eta$ ’s state has been clamped to the pre-fixed vector  $\vec{v}$ . The state to which  $\eta$  is clamped can be chosen from  $\eta$ ’s possible states, or it can be an arbitrary vector. In the latter case, care must be taken to ensure that  $G$  is defined over – or extended to – the “illegal” clamped states. Figure 2 shows an example of clamping when the state of each of the four agents is a unary action vector for each time step.

Now, let us define a family of payoff functions, called **Wonderful Life (WL)** utility functions. For each agent  $\eta$ , the WL utility parameterized by the vector  $\vec{v}$  to which agent  $\eta$  is clamped is given by:

$$WLU_{\eta} \equiv G(\zeta) - G(\zeta_{\eta}, CL_{\eta}^{\vec{v}}). \quad (6)$$

Note that, regardless of the choice of the clamping parameter, WL utilities are of the form given in Equation 4, and therefore are factored. Furthermore, it can be proven that in many circumstances, especially in large problems,  $\lambda_{\eta, WLU}(\zeta) \geq \lambda_{\eta, G}(\zeta)$ , i.e., WLU has higher differential learnability than does the team game choice of payoff utilities [31]. This is mainly due

to the second term of WLU, which removes a lot of the effect of other agents (i.e., noise) from  $\eta$ 's utility. The result is that though WLU does not theoretically match the high learnability of AU, in practice convergence to optimal  $G$  with WLU is much quicker (up to orders of magnitude so [31, 32]) than with a team game.

Though all WL payoff utilities are factored regardless of the choice of  $\vec{v}$ , the selection of  $\vec{v}$  affects the learnability of the payoff function. Therefore, in practice matching the proper clamping parameter to the domain greatly improves the performance of the system [31]. In many circumstances there is a particular choice of clamping parameter for agent  $\eta$  that is a “null” move (i.e.,  $\vec{v} = \vec{0}$ ) for that agent, equivalent to removing that agent from the system, hence the name of this payoff function. For such a clamping parameter WLU is closely related to the economics technique of “endogenizing a player’s (agent’s) externalities” [15]. Indeed, WLU has conceptual similarities to Vickrey tolls [23] in economics, and Groves’ mechanism [6] in mechanism design<sup>5</sup>. However, because WLU can be applied to arbitrary, time-extended utility functions, and need not be restricted to the “null” clamping operator interpretable in terms of “externality payments”, it can be viewed a generalization of these concepts.

For example, it is usually the case that using WLU with a clamping parameter that is as close as possible to the expected action (denoted by  $\vec{a}$ ), and not the “null” action, results in higher learnability than does clamping to the null move. For example, in Figure 2, if the probabilities of agent 2 making each of its possible moves is 1/3, then one can expect that setting the clamping parameter to  $\vec{a}$  would result in maximizing learnability. Accordingly, in practice, use of such an alternative WLU derived as a “mean-field” approximation<sup>6</sup> to AU almost always results in better values of  $G$  than does the “endogenizing” WLU [32].

Furthermore, though a conceptual similarity between AU and the AGV-Arrow mechanism [4] appears to exist, there is a fundamental difference: In the AGV-Arrow mechanism, one averages the actions of “other” players conditioned on one’s own action, i.e., one gets a sense of the “average background”. This is fundamentally different than the operation performed by AU which is to average one’s own actions, based on others’ actions<sup>7</sup>, and subtract the resulting world utility from the actual world utility. This provides a difference utility that measures the “impact” of taking a particular action, given everyone else’s state.

Intuitively, one can look at AU and WLU from the perspective of a human company, with  $G$  the “bottom line” of the company, the agents  $\eta$  identified with the employees of that company, and the associated  $g_\eta$  given by the employees’ performance-based compensation packages. For example, for a “factored company”, each employee’s compensation package contains incentives designed such that the better the bottom line of the corporation, the greater the employee’s compensation. As an example, the CEO of a company wishing to have the payoff utilities of the employees be factored with  $G$  may give stock options to the employees. The net effect of this

<sup>5</sup>Note also that Groves’ mechanism is restricted to public resources where an agent’s use of that resource does not affect the ability of other agents to use the resource (e.g., a lighthouse) and therefore the tragedy of the commons does not arise. Grove’s mechanism was especially formulated to solve the problem of agents being untruthful in reporting their utility for public goods, a problem not present in the collectives framework.

<sup>6</sup>Formally, our approximation is exact only if the expected value of  $G$  equals  $G$  evaluated at the expected joint move (both expectations being conditioned on given moves by all agents other than  $\eta$ ). In general though, for relatively smooth  $G$ , we expect such a mean-field approximation to AU, to give good results, even if the approximation does not hold exactly [32].

<sup>7</sup>In this analogy, we equate state to action, e.g., an agent’s action uniquely determines its state.

action is to ensure that what is good for the employee is also good for the company. In addition, if the compensation packages have “high learnability”, the employees will have a relatively easy time discerning the relationship between their behavior and their compensation. In such a case the employees will both have the incentive to help the company and be able to determine how best to do so. Note that in practice, providing stock options is usually more effective in small companies than in large ones. This makes perfect sense in terms of the collectives formalism, since such options generally have higher learnability in small companies than they do in large companies, in which each employee has a hard time seeing how his/her moves affect the company’s stock price.

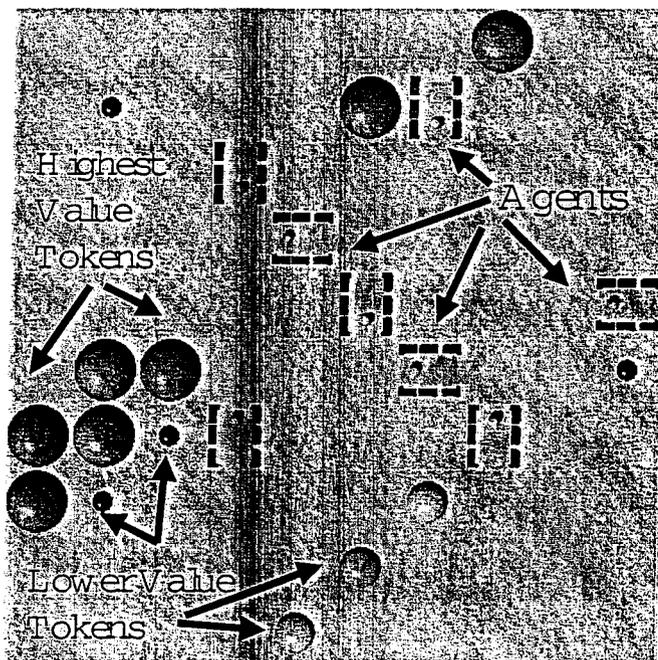


Figure 3: Agents collecting tokens of varying value.

### 3 Multi-agent Grid World Problem

#### 3.1 Problem Description

A common reinforcement learning problem is the Grid World Problem [20], where an agent navigates about a two-dimensional  $n \times n$  grid. At each time step, the agent can move up, down, right or left one grid square, and receives a reward after each move. The observable state space for the agent is its grid coordinate and the reward it receives depends on the grid square to which it moves. In the episodic version, which is the focus of this paper, the agent moves for a fixed number of time steps, and then is returned to its starting location. This problem typically requires the use of a reinforcement learner that can optimize a sum of rewards in contrast to one that optimizes an immediate reward, since the agent may have to cross squares of low reward value to enter the squares of high value. Q-learners or the Sarsa algorithm [20] are often used

for this problem.

In this paper we apply the theory of collectives to a multi-agent version of the grid world problem. In this instance of the problem there are multiple agents navigating the grid simultaneously interacting with each others' rewards. This reward interaction is modeled through the use of tokens that are distributed throughout the grid squares of the gridworld (Figure 3). Each token has a value between zero and one, and each grid square can have at most one token. When an agent moves into a grid square it receives a reward for the value of the token and then removes the token so that a reward will no longer be received when another agent enters the grid square. However, all the tokens are reset at the end of an episode. The global objective of the Multi-agent Grid World Problem is to collect the highest aggregated value of tokens in a fixed number of time steps.

The Multi-agent Grid World Problem is an idealized version of many real world problems, including the control of multiple planetary exploration vehicles (e.g., rovers on the surface of Mars, collecting rocks in an attempt to maximize total scientific return, submersible under Europa examining potential life signs). Furthermore, the agent interaction provides a critical study of coordination and interference, as the agents have the potential to work at cross-purposes. This problem is particularly interesting in a multi-agent setting because each agent attempting to maximize the value of the tokens it collects, can drive the world utility to severely sub-optimal values. As such, the design of the payoff functions is crucial in this problem, and we address this issue below.

### 3.2 Collective-Based Solution

To pose the Multi-agent Grid World Problem in the form of the collective framework we need to define:

- $L_{\eta,t}$ : The matrix representing the location of an agent. If agent  $\eta$  at time  $t$  is in location  $(x, y)$ , then  $L_{\eta,t,x,y} = 1$ ; otherwise  $L_{\eta,t,x,y} = 0$ . Furthermore,  $\{L_{\eta,t}\}$  denotes the set all the agents' location matrices.
- $L_{\eta,t}^a$ : The location matrix agent  $\eta$  would have had at time  $t$ , had it taken action  $a$  at time step  $t - 1$ .
- $L_{\eta}$ : The location matrix of agent  $\eta$  across all time ( $L_{\eta} = \sum_t L_{\eta,t}$ ).
- $L_{\eta,<t}$ : The location matrix of agent  $\eta$  across times less than  $t$  ( $L_{\eta,<t} = \sum_{t'<t} L_{\eta,t'}$ ).
- $L_t$ : The location matrix of all agents at time  $t$  ( $L_t = \sum_{\eta} L_{\eta,t}$ ).
- $L$ : The location matrix of all agents across all time ( $L = \sum_t L_t = \sum_t \sum_{\eta} L_{\eta,t}$ ).
- $L_{<t}$ : The location matrix of all agents across times less than  $t$  ( $L_{<t} = \sum_{t'<t} L_{t'} = \sum_{t'<t} \sum_{\eta} L_{\eta,t'}$ ).
- $L_{\neg\eta}$ : The location matrix of all agents other than  $\eta$  across all time ( $L_{\neg\eta} = L - L_{\eta}$ ).
- $L_{\neg\eta,<t}$ : The location matrix of all agents other than  $\eta$  across times less than  $t$  ( $L_{\neg\eta,<t} = L_{<t} - L_{\eta,<t}$ ).

- $\Theta$ : The initial token value matrix (e.g.,  $\Theta_{x,y}$  contains the initial value of the token at location  $(x,y)$ ).

The space  $\mathbf{Z}$  is composed of  $\Theta$  and the set of all possible location matrices,  $\{L_{\eta,t}\}$ , given the length of an episode. A worldline  $\zeta$  is a point in this space, i.e., the combination of the token configurations  $\Theta$ , along with a particular set  $\{L_{\eta,t}\}$ . We now define the function  $V(L, \Theta)$  which returns the value of a token received from a location matrix. Formally:

$$V(L, \Theta) = \sum_{x,y} \Theta_{x,y} \min(1, L_{x,y}). \quad (7)$$

The global utility  $G(\zeta)$  is the sum off all the tokens collected during an episode:

$$G(\zeta) = V(L, \Theta). \quad (8)$$

Based on the definitions and world utility given above, let us now derive the collective-based utility functions for this domain. In this formulation, the AU (given in Equation 5) become:

$$AU_{\eta}(\zeta) = G(\zeta) - \sum_{\bar{a} \in \bar{A}_{\eta}} p_{\bar{a}} V(L_{\eta} + L_{\eta}^{\bar{a}}, \Theta) \quad (9)$$

where  $A_{\eta}$  is the set of possible action sequences agent  $\eta$  can take. The second term in the equation is the expected value of the global utility over all the possible actions of agent  $\eta$ .

Now, let us formulate the WL utilities for this domain. First, setting the clamping parameter  $CL_{\eta}$  to the null vector, we obtain WL utility where the agent is removed from the worldline:

$$WLU_{\eta}^{\bar{0}}(\zeta) = G(\zeta) - V(L_{\eta}, \Theta). \quad (10)$$

This utility returns an agent’s contribution to the world utility. Note, this utility differs from one where the values of the tokens present in the locations visited by the agent are summed (i.e., a selfish utility).  $WLU_{\eta}^{\bar{0}}$  gives the value of the tokens *in locations not visited by other agents*, i.e., the values of token that would not have been picked up had agent  $\eta$  not been in the system.

Next, let us define the WL utility resulting from agent  $\eta$  taking the fictitious average action, where it partially takes all possible actions:

$$WLU_{\eta}^{\bar{a}}(\zeta) = G(\zeta) - V(L_{\eta} + \sum_{\bar{a} \in \bar{A}_{\eta}} p_{\bar{a}} L_{\eta}^{\bar{a}}, \Theta) \quad (11)$$

Because these utilities are based on the performance on a full episode, they are problematic to work with directly. We therefore introduce single time step “rewards” that will help in learning the set of actions (e.g., through Q-learners or Sarsa learners) that will lead to good values for the utility. Note, that the utilities will be undiscounted sums of these rewards. To that end, first let us decompose an arbitrary utility  $U$  in the following manner:

$$U(L) = \sum_t U(L_{<t+1}) - U(L_{<t}). \quad (12)$$

Now, let us define the single time step reward  $R_t$  by:

$$R_t(L) = U(L_{<t+1}) - U(L_{<t}) \quad (13)$$

Now we can generate the four single time step reward versions of the four utilities<sup>8</sup>:

$$GR_t(\zeta) = V(L_{<t+1}, \Theta) - V(L_{<t}, \Theta) \quad (14)$$

$$AR_{\eta,t}(\zeta) = GR_t(\zeta) - \sum_{\bar{a} \in \bar{A}_\eta} p_{\bar{a}} (V(L_{\eta,<t+1} + L_{\eta,<t+1}^{\bar{a}}, \Theta) - V(L_{\eta,<t} + L_{\eta,<t}^{\bar{a}}, \Theta)) \quad (15)$$

$$WLR_{\eta,t}^{\bar{0}}(\zeta) = GR_t(\zeta) - (V(L_{\eta,<t+1}, \Theta) - V(L_{\eta,<t}, \Theta)) \quad (16)$$

$$WLR_{\eta,t}^{\bar{a}}(\zeta) = GR_t(\zeta) - \left[ V \left( L_{\eta,<t+1} + \sum_{\bar{a} \in \bar{A}_\eta} p_{\bar{a}} L_{\eta,<t+1}^{\bar{a}}, \Theta \right) - V \left( L_{\eta,<t} + \sum_{\bar{a} \in \bar{A}_\eta} p_{\bar{a}} L_{\eta,<t}^{\bar{a}}, \Theta \right) \right] \quad (17)$$

Note that as expressed above the formulation for  $AR$  and  $WLR^{\bar{a}}$  has significant drawbacks. First, the set of all possible action sequences is very large, and grows exponentially with  $t$ . Second, without prior information, the average path contains little information and for  $WLR$  is similar to clamping to zero. To side-step these issues, we make the approximation that each action is equally likely, and compute the average action over the actions available to the agent in the previous time step:

$$AR_{\eta,t}(\zeta) = GR_t(\zeta) - \sum_{a \in A_{\eta,t-1}} p_a (V(L_{\eta,<t+1} + L_{\eta,<t+1}^a, \Theta) - V(L_{\eta,<t} + L_{\eta,<t}^a, \Theta))$$

$$WLR_{\eta,t}^a(\zeta) = GR_t(\zeta) - \left[ V \left( L_{\eta,<t+1} + \sum_{a \in A_{\eta,t-1}} p_a L_{\eta,<t+1}^a, \Theta \right) - V \left( L_{\eta,<t} + \sum_{a \in A_{\eta,t-1}} p_a L_{\eta,<t}^a, \Theta \right) \right]$$

Note, the average action sequence has been replaced with the sequence of average actions, that is the sequence where at each time step the average action has been taken. Because of the arbitrariness of the clamping operator (see discussion in Section 2) and the fact that theoretically, clamping to any fixed vector results in a factored system, this approximation is milder for  $WL^a$  than it is for  $AU$ .

## 4 Experimental Results

To evaluate the effectiveness of the collective-based approach in the Multi-agent Grid World, we conducted experiments on three different types of token distributions. The payoff utility function

<sup>8</sup>In the actual implementation there are some tie breaking rules if more than one agent goes into the same square at the same time.

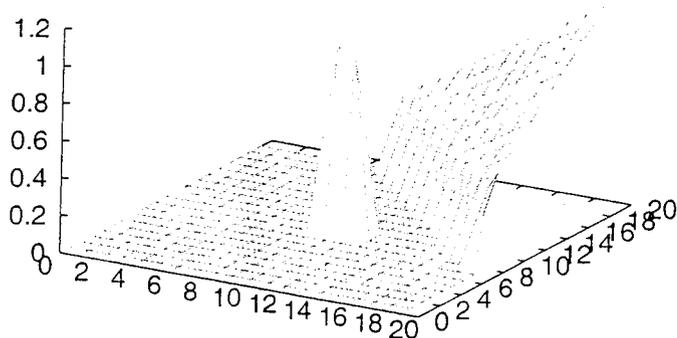


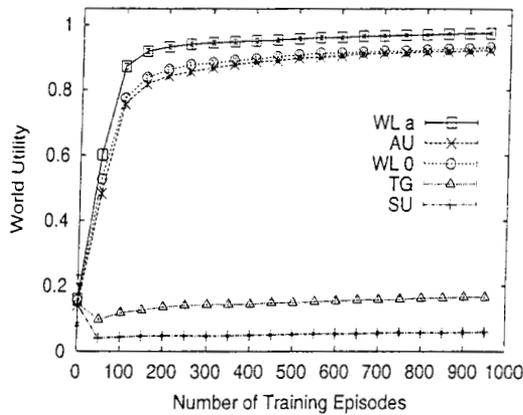
Figure 4: Distribution of Token Values in “Corner” World

investigated included: (i) the Selfish Utility (SU), where each agent receives the weighted total of the tokens that it alone collected. It is the natural extension of the single agent problem, and represents the optimal utility in the single rover domain; (ii) the Team Game (TG) utility where each agent received the full world utility; (iii) the  $WL^{\bar{0}}$  utility, where there clamping parameter is set to  $\bar{0}$ . Intuitively, this utility computes the contribution an agent makes to the token collection, by looking at the difference in the total token collection with and without that agent; (iv) the  $WL^{\bar{a}}$  utility, where there clamping parameter is set to  $\bar{a}$ , representing the difference between in utility value resulting from an agent’s actual action and its “smeared” action; and (v) the AU, where the agent’s contribution is computed as the difference between the action it took and its expected action.

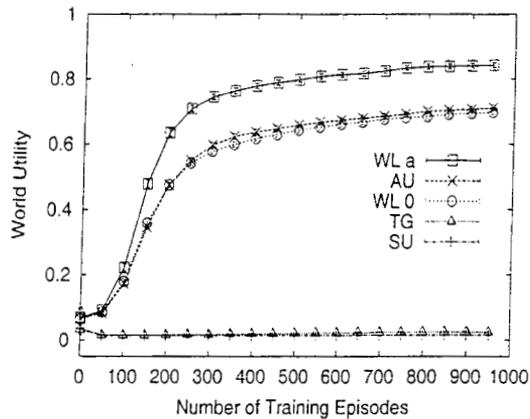
In the following subsections we evaluate the five payoff utilities using three different distributions of tokens. Since we are unaware of any standard gridworld benchmarks that specify reward distributions, we created artificial distributions of tokens that could illustrate the important collective learning issues. The first set of tokens is the most “unnatural,” but requires the agents to optimize a sum of rewards instead of an immediate reward, while working together, if high world utility is to be achieved. The second set of tokens has similar properties to the first set, but has smoother transitions in token values. The final set is randomly generated from Gaussian kernels on every trial, to illustrate that the collective-based principles still hold on less hand crafted token distributions.

#### 4.1 Corner World Token Value Distribution

The first experimental domain we investigated consisted of a world where where the “highly valued” tokens are concentrated in one corner, with a second concentration near the center where the rovers are initially located. The rest of the world is uniformly filled with tokens of little importance. Figure 4 conceptualizes this distribution for a 20x20 world.



(a) 40 Rovers on a 20x20 grid.



(b) 100 Rovers on a 32x32 grid.

Figure 5: Effect of Payoff Utility on System Performance.

Figure 5(a) shows the performance of the different payoff utilities for 40 agents on a 400 unit-square world for the token value distribution shown in Figure 4, and where an episode consists of 20 time steps (error bars of  $\pm$  one  $\sigma$  are included, though they are smaller than the symbols). The performance measure in these figures is “normalized” world utility given by  $\frac{V(L, \Theta)}{V(1_n, \Theta)}$ , where  $1_n$  is the  $n \times n$  matrix of ones. This normalized utility provides the fraction of token values that was collected by the agents (a value of 1 means all available tokens were collected).

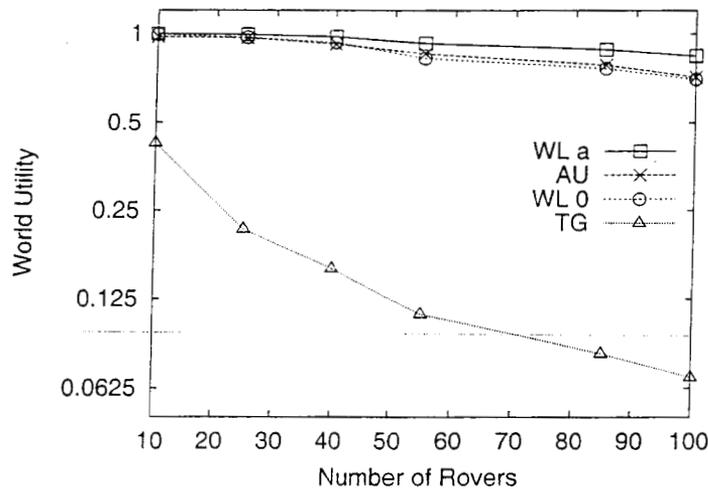


Figure 6: Scaling Properties of Different Payoff Functions.

The results show that SU produced poor results, results that were indeed worse than random actions. This is caused by all agents aiming to acquire the most valuable tokens, and congregating towards the corner of the world where such tokens are located. In essence, in this case agents using the SU payoff competed, rather than cooperated with one another. The agents using TG

fared marginally better, but their learning was slow. This system was plagued by the signal-to-noise problem associated with each agent receiving the full world reward for each individual action they took. Notice both the selfish agents and those trained with TG had a drop in their performance in the early going, as they learned the “wrong” actions (as far as the world utility is concerned). Agents using TG payoffs overcame this early setback whereas selfish agents never did. In contrast, agents using  $WL^{\bar{0}}$  and AU performed very well, and agents using  $WL^{\bar{a}}$  performed almost optimally. In each of these three cases, the reinforcement signal the agents received was both factored and showed how their actions affected the world reward more clearly than did the TG reinforcement signal.

For a scaled up version of the same token value distribution, Figure 5(b) shows the results for 100 agents on a 1024 unit-square grid, where an episode consists of 32 time steps. Qualitatively, the results are similar to the 40 agent case. However, note that the team game agents have a harder time learning, because in this case the reinforcement signal is even further diluted. Furthermore, the performance of  $WL^{\bar{a}}$  is now clearly superior to that of  $WL^{\bar{0}}$ , showing that using the degree of freedom given by the clamping parameter provides significant improvements over solutions aimed at “endogenizing externalities” or similar to the Groves mechanism ( $WL^{\bar{0}}$ ).

Figure 6 explores the scaling issue in more detail. As the number of agents was increased, we kept the difficulty of the problem the same by increasing the size of the gridworld, and allocating more time for an episode. Specifically the ratio of the number of agents to total number of grid squares and the ratio of the number of agents to total value of tokens was held constant. In addition the ratio of the furthest grid square from the agents’ starting point to the total amount of time in an episode was also held constant. The scaling results show that agents using TG payoffs were not hampered as much by the noise associated with other agents when the number of agents was low. As the system scaled up however their performance deteriorated rapidly. Agents using WL and AU payoffs on the other hand were not strongly affected by the increase in the size of the problem. This underscores the need for a payoff utility function that has good signal-to-noise properties so that in large systems, the agents have an opportunity to learn the actions that will optimize their payoff utilities.

## 4.2 Incline World Token Value Distribution

In the second experimental domain we investigated a world where the “highly valued” tokens are still concentrated in one corner, but where there is a “ridge” of moderately high values along a side, starting from the opposite corner. The actual distribution for the token values  $\Theta_{x,y}$  on a two dimensional (x,y) map is given by:

$$\Theta_{x,y} = 1.0 - \left( \frac{x}{s} \left( 1.0 - \frac{s-y}{s} \right) \right) - \left( 0.5 \frac{s-y}{s} \right) \quad (18)$$

where  $s$  is the one dimensional “size” of the map (i.e.,  $s = 20$  for 40 agents, and  $s = 32$  for 100 agents). Figure 7 conceptualizes this distribution for a 10x10 world ( $s=10$ ).

Figure 8 shows the performance of the different payoff utilities for 40 agents and 100 agents on 400 and 1024 unit-square worlds, respectively, on the token value distribution given by Equation 18.

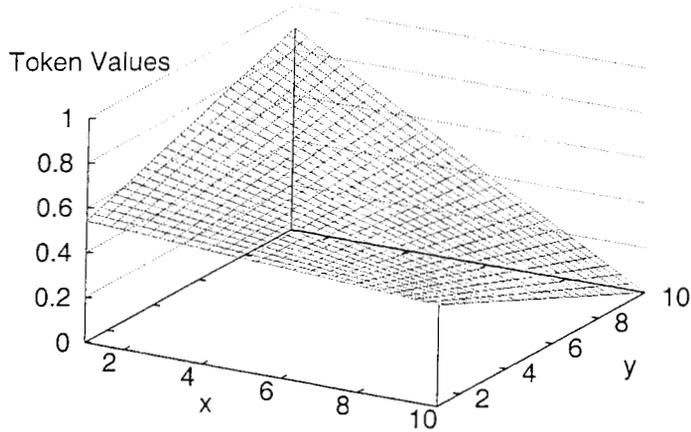
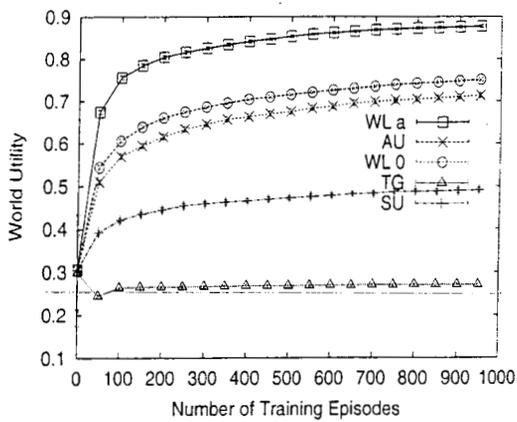
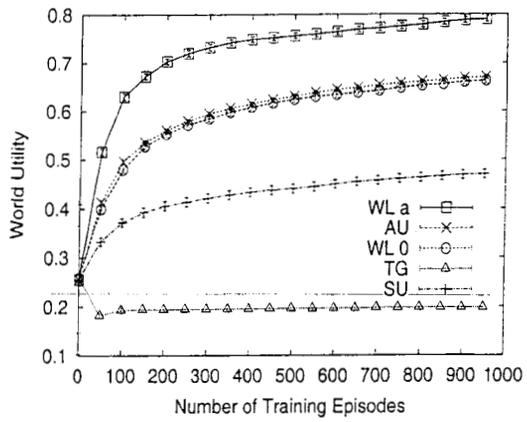


Figure 7: Distribution of Token Values in "Incline" World



(a) 40 Rovers on a 20x20 grid.



(b) 100 Rovers on a 32x32 grid.

Figure 8: Effect of Payoff Utility on System Performance for Incline World Token Value Distribution.

The results show that the WL payoff is unaffected by the change in the token value distribution. Both TG and SU payoffs in contrast perform better in this case, showing a much larger sensitivity to the token distribution. The improvements in SU are easily explained: The area surrounding the high token values contains sufficiently many tokens that even when the SU agents are all trying to reach the high valued tokens, they help the world utility.

Though agents using TG collect a larger portion of token as compared to the previous token configuration, the lack of improvement displayed in the system where agents use TG payoffs is noteworthy. Because of the noise in the system, these agents do not even learn to “walk” in the right direction in the allotted number of episodes.

### 4.3 Random World Token Value Distribution

In the final set of experiments, we investigate the behavior of agents in a gridworld where the token values are randomly distributed. In this world, for  $N$  agents, there are  $N/3$  Gaussian ‘attractors’ whose centers are randomly distributed. Figure 9 shows an instance of the gridworld using this distribution for the 20x20 world, used in the experiments with 40 agents.

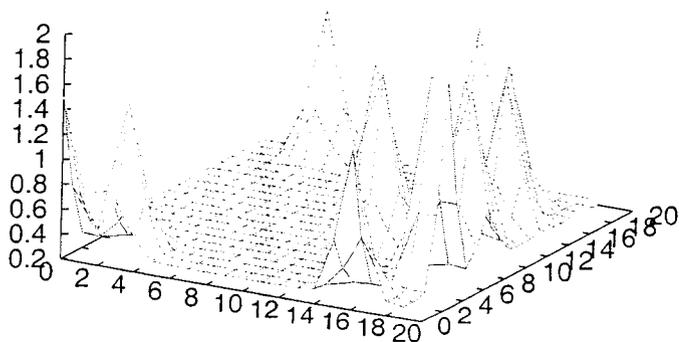
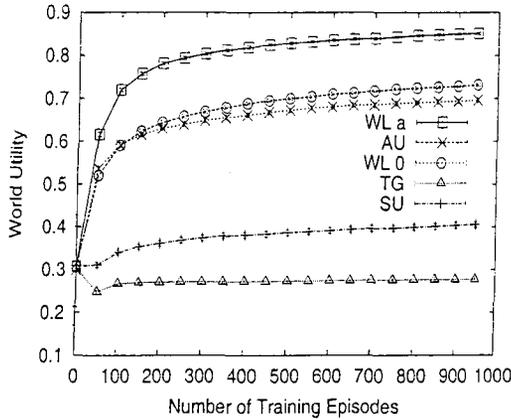


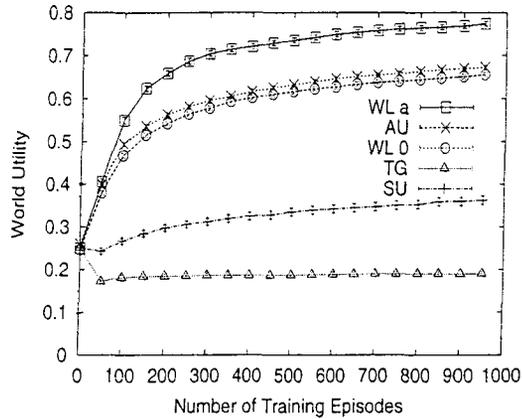
Figure 9: Distribution of Token Values in “Random” World

Figure 10 shows that the performance of the agents in the “random” world are very similar to the “incline” world, but for the poorer performance of the SU payoff function. This can be explained by the token value distributions: there are many “dry” patches, and agents aiming for the high valued token do not necessarily get the consolation of mid-valued tokens. The WL payoff again does well for both clamping parameters.

This experiment illustrates that when agents need to use a “divide and conquer” approach, the selfish utility performs poorly. Furthermore, these experiments illustrate that the collective-based payoff functions are superior across a wide range of token distributions ranging from smooth to irregular to random.



(a) 40 Rovers on a 20x20 grid.



(b) 100 Rovers on a 32x32 grid.

Figure 10: Effect of Payoff Utility on System Performance for Random World Token Value Distribution.

## 5 Nash Equilibrium and World Utility Optimum

Figure 11 illustrates a simple example with two agents, a six square world, and where each agent can choose to move left or right for two time steps. There are two tokens, one of values 5 and the other of value 10 that the agents can pick up by entering the appropriate square.

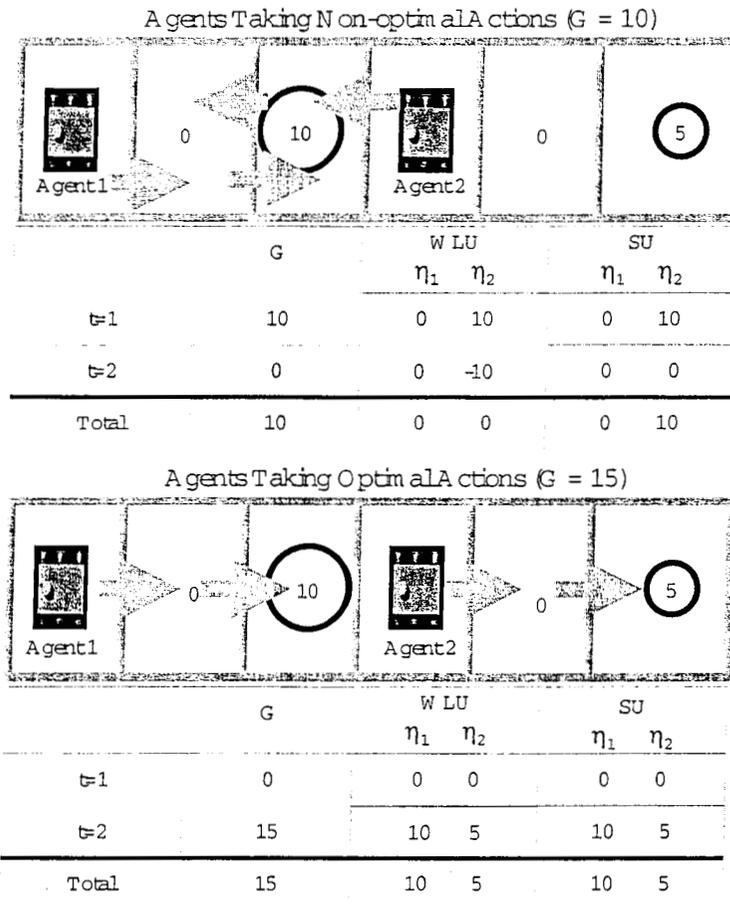
Consider the joint set of moves where agent 1 moves right twice, and agent 2 moves left twice<sup>9</sup>. In this scenario agent 2 picks up a token worth 10 units on the first time step and nothing. Agent 1 does not pick up any tokens. Figure 11 summarizes the reward and utility values associated with this move. Agent 1 receives an SU of 10 (10 for the first step, 0 for the second) whereas agent 2 receives an SU of 0. This results in a world reward of 10 for the first time step and 0 for the second, resulting in a world utility of 10. Incidentally, this is the solution the system settles into if the agents are indeed using SU as their payoff utilities. For SU, this is a Nash Equilibrium: There is no unilateral moves that a player can make that will improve its utility.

Now, let us analyze the  $WL^{\bar{0}}$  payoff utility for agent 2 for this set of moves:<sup>10</sup> For the first time step, the WL reward is the same as SU: agent 2 receives 10 for picking up the token. It is in the second time step though the differences emerges: The first term of WL as given in Equation 17 (i.e., the world reward for time step 2) is 0 for this time step as no tokens are picked up. However, in the worldline where agent 2 has been clamped to zero, the first parameter of the  $V$  function,  $L_{\eta}$ , does not include the locations of agent 2, causing this function do disregard any tokens agent 2 previously picked up. This causes agent 2 to credit agent 1 for picking up the token of value 10 in the second time step. Agent 2 then computes a value of 10 for the world reward of this state where it's clamped its action to  $\bar{0}$ . The WL reward for agent 2 for this time

<sup>9</sup>The second step of agent 2 is arbitrary.

<sup>10</sup>Both AU and WL with clamping to  $\bar{a}$  provide similar results, but we omit them for clarity.

step is then given by:  $WLR_{\eta_2, t=2} = 0 - 10 = -10$ .



**Figure 11: WLU Nash Equilibria and World Utility Optima**

Now the time-extended WLU for agent 2 can be computed by summing the WLRs. This results in a WLU of 0, even though agent 2 picks up a token weighted 10 (10 for  $t=1$  and -10 for  $t=2$ ). The interpretation for this “counter-intuitive” utility value is clear: because that token would have been picked up by agent 1 at another time step, the net effect of agent 2’s actions on the world utility was nil, resulting in a WLU value of 0 for agent 2 for this set of actions. Because moving to the right twice provides a WLU value of 5 for agent 2 (as detailed in Figure 11), an agent optimizing its WL payoff utility will take this second action. Similarly agent 1 moving right twice will receive a WLU of 10. As this simple example shows, each agent maximizing its WLU leads the system to the world utility maximum where both tokens are picked up.

Let us analyze the game-theoretic equilibrium states for WLU and SU in these two solutions: The SU is in a Nash equilibrium for the first set of moves, in that neither agent can improve its SU by unilaterally changing its actions. Therefore, the system is “stuck” in this suboptimal solution. Furthermore, even if the agents stumble upon the second solution by accident, they will not remain there, as this solution is unstable with payoff utilities given by SU: Agent 2 can change its move (in future episodes) and improve its payoff utility from 5 to 10. That this move reduces agent 1’s utility from 10 to 0, and the world utility from 15 to 10 has no influence on

agent 2’s actions. Furthermore, this solution is also Pareto-optimal, in that there is no set of joint moves that improves the utility of *both* agents. This example shows a simple case where Pareto optimality and optimum of the world utility do not necessarily coincide<sup>11</sup>, and simply seeking a Pareto-optimal solution will not necessarily lead to high values of the world utility function.

On the other hand agents whose payoff utilities are given by the WLU are in a Nash Equilibrium in the second set of actions. They will therefore seek this solution as it offers higher payoff utilities for each agent. The use of WLU has the net effect of “aligning” the Nash equilibrium of the agents with the world utility optimum, ensuring that when the agents optimize their payoff utilities, the world utility is also at a local – and in this case also the global – optimum.

## 6 Discussion

In this work we focus on the problem of designing a collective of autonomous agents that individually learn sequences of actions such that the resultant sequence of joint actions achieves a predetermined global objective. In particular we discuss the problem of controlling multiple agents in a gridworld, a problem related to many real world problems including exploration vehicles trying to maximize aggregate scientific data collection (e.g., rovers on the surface of Mars).

This paper illustrates how the theory of collectives can be used to leverage the work done on existing reinforcement learners that are able to work with sequences of actions, so that they can be extended to multi-agent problems. In this domain, we addressed the critical issue of what utility functions those agents should strive to maximize. We extended previous results on collective intelligence to agents attempting to maximize sequences of actions, and used Q-learning with rewards set by the theory of collectives. Our results demonstrate that agents using collective-derived goals outperform both “natural” extensions of single agent algorithms and global reinforcement learning solutions based on “team games”.

Even the simplest collective-derived utility,  $WL^{\bar{0}}$ , showed marked improvement over greedy and team game utilities as it was able to scale well, while still directing agents to “work together.” To try to increase performance further, this paper presented two utilities in addition to  $WL^{\bar{0}}$ :  $WL^{\bar{a}}$  and AU. While  $WL^{\bar{0}}$  performed well in these problems,  $WL^{\bar{a}}$  provided further improvements, and approached the optimal solution in many cases. This improvement was due to  $WL^{\bar{a}}$  returning an agents contribution compared to an average action rather than to the more extreme case comparing it to no action at all. The experimental results confirm the theoretical analysis that shows  $WL^{\bar{a}}$  having a higher learnability than  $WL^{\bar{0}}$ .

While  $WL^{\bar{a}}$  proved to be a superior utility, our investigations revealed an interesting situation where AU, the theoretically “best” strategy, was not necessarily the best approach in practice. Although AU is theoretically superior to WLU (higher learnability), two issues prevent us from fully exploiting its power: First, the “expected” action is impossible to compute in a time extended setting, since even a simple case where an agent has four actions and ten time steps leads to  $4^{10}$  possible actions. Even Monte Carlo sampling of such a space will yield highly

<sup>11</sup>Note, in this case the world utility optimum is also a Pareto-optimal, but there are no guarantees that the system will stumble upon the “desirable” Pareto optimal solution.

inaccurate estimates of the potential actions and their rewards. Second, estimating the correct probability distributions over the possible actions causes the utility values to change, creating a self-consistency problem. To sidestep both issues, in this article we chose to focus on the last time step (e.g., current step for the agent) and approximated the AU with the agent taking each of the four actions possible in that time step with equal likelihood. The resulting utility function provided good solutions, but the performance of such a “handicapped” AU did not exceed those of the conceptually simpler  $WL^{\bar{0}}$  and was well below that of  $WL^{\bar{a}}$ .

Finally, a game-theoretic analysis of the utilities provided by the theory of collectives sheds light on the reasons for the dramatic improvements obtained over selfish utilities and team games: While the Nash equilibrium of the system in which each agent pursues a selfish goal does not correspond to a globally good solution, the global optimum is indeed a Nash equilibrium of the system in which agents use collective-based utilities. Furthermore, such utilities provide “good” off-equilibrium signals to lead the system into desirable equilibrium states, whereas systems in which all agents use team game utilities fail to reach the desirable states (e.g., Nash equilibrium states which also are optima of the world utility) due to the excessive “noise” in the system.

**Acknowledgements:** The authors thank David Wolpert for his contributions and insightful comments.

## References

- [1] J. A. Boyan and M. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. In *Advances in Neural Information Processing Systems - 6*, pages 671–678. Morgan Kaufman, 1994.
- [2] J. A. Boyan and A. Moore. Learning evaluation functions for global optimization and boolean satisfiability. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*. AAAI Press, 1998.
- [3] R. H. Crites and A. G. Barto. Improving elevator performance using reinforcement learning. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems - 8*, pages 1017–1023. MIT Press, 1996.
- [4] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, Cambridge, MA, 1991.
- [5] A. Greenwald, E. Friedman, and S. Shenker. Learning in network contexts: Experimental results from simulations. *Journal of Games and Economic Behavior: Special Issue on Economics and Artificial Intelligence*, 35(1/2):80–123, 2001.
- [6] T. Groves. Incentives in teams. *Econometrica*, 41:617–631, 1973.
- [7] G. Hardin. The tragedy of the commons. *Science*, 162:1243–1248, 1968.
- [8] J. Hu and M. P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 242–250, June 1998.

- [9] J. Hu and M. P. Wellman. Online learning about other agents in a dynamic multiagent system. In *Proceedings of the Second International Conference on Autonomous Agents*, pages 239–246, May 1998.
- [10] B. A. Huberman and T. Hogg. The behavior of computational ecologies. In *The Ecology of Computation*, pages 77–115. North-Holland, 1988.
- [11] N. R. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1:7–38, 1998.
- [12] M. Kearns and D. Koller. Efficient reinforcement learning in factored MDPs. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 740–747, 1999.
- [13] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning*, pages 157–163, 1994.
- [14] D. Monderer and L. S. Sharpley. Potential games. *Games and Economic Behavior*, 14:124–143, 1996.
- [15] W. Nicholson. *Microeconomic Theory*. The Dryden Press, seventh edition, 1998.
- [16] T. Sandholm and R. Crites. Multiagent reinforcement learning in the iterated prisoner’s dilemma. *Biosystems*, 37:147–166, 1995.
- [17] T. Sandholm, K. Larson, M. Anderson, O. Shehory, and F. Tohme. Anytime coalition structure generation with worst case guarantees. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 46–53, 1998.
- [18] S. Sen. *Multi-Agent Learning: Papers from the 1997 AAAI Workshop (Technical Report WS-97-03)*. AAAI Press, Menlo Park, CA, 1997.
- [19] P. Stone and M. Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3), 2000.
- [20] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.
- [21] K. Sycara. Multiagent systems. *AI Magazine*, 19(2):79–92, 1998.
- [22] K. Tumer and D. H. Wolpert. Collective intelligence and Braess’ paradox. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 104–109, Austin, TX, 2000.
- [23] W. Vickrey. Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961.
- [24] C. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3/4):279–292, 1992.
- [25] G. Weiss, editor. *Multiagent Systems*. MIT Press, Cambridge, MA, 1999.

- [26] M. P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. In *Journal of Artificial Intelligence Research*, 1993.
- [27] D. H. Wolpert. The mathematics of collective intelligence. pre-print, 2001.
- [28] D. H. Wolpert, E. Bandari, and K. Tumer. Improving simulated annealing by recasting it as a non-cooperative game. 2001. submitted.
- [29] D. H. Wolpert, S. Kirshner, C. J. Merz, and K. Tumer. Adaptivity in agent-based routing for data networks. In *Proceedings of the fourth International Conference of Autonomous Agents*, pages 396–403, 2000.
- [30] D. H. Wolpert, J. Sill, and K. Tumer. Reinforcement learning in distributed domains: Beyond team games. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 819–824, Seattle, WA, 2001.
- [31] D. H. Wolpert and K. Tumer. An Introduction to Collective Intelligence. Technical Report NASA-ARC-IC-99-63, NASA Ames Research Center, 1999. URL:[http://ic.arc.nasa.gov/ic/projects/coin\\_pubs.html](http://ic.arc.nasa.gov/ic/projects/coin_pubs.html). To appear in Handbook of Agent Technology, Ed. J. M. Bradshaw, AAAI/MIT Press.
- [32] D. H. Wolpert and K. Tumer. Optimal payoff functions for members of collectives. *Advances in Complex Systems*, 4(2/3):265–279, 2001.
- [33] D. H. Wolpert, K. Tumer, and J. Frank. Using collective intelligence to route internet traffic. In *Advances in Neural Information Processing Systems - 11*, pages 952–958. MIT Press, 1999.
- [34] D. H. Wolpert, K. Wheeler, and K. Tumer. Collective intelligence for control of distributed dynamical systems. *Europhysics Letters*, 49(6), March 2000.